# Is GENERIC_ALL equivalent to GENERIC_READ | GENERIC_WRITE | GENERIC_EXECUTE?

devblogs.microsoft.com/oldnewthing/20170310-00

March 10, 2017

Raymond Chen

A customer wanted to know whether passing `GENERIC_ALL` as an access mask is effectively equivalent to passing `GENERIC_READ | GENERIC_WRITE | GENERIC_EXECUTE`. Specifically, they were interested in the answer to this question with respect to the `CreateFile` function.

Okay, first question first. Is `GENERIC_ALL` effectively equivalent to `GENERIC_READ | GENERIC_WRITE | GENERIC_EXECUTE`?

The answer is “It depends.”

Each object decides what these generic access masks mean. Now, the intended use is that `GENERIC_READ` correspond to whatever “read” access means for an object, `GENERIC_WRITE` correspond to whatever “write” access means for an object, and `GENERIC_EXECUTE` correspond to whatever “execute” access means for an object. It’s also the intended use that `GENERIC_ALL` represent whatever access makes the most sense for “all access”.

But that’s just the intended use. There is nothing physically preventing an object from giving those four generic access masks nonsensical values. Because anybody can make up a generic mapping. Therefore, there’s nothing you can guarantee about the relationship between the generic access masks beyond “they are what the object decides they are.”

In practice, `GENERIC_ALL` is at least as big as `GENERIC_READ | GENERIC_WRITE | GENERIC_EXECUTE`, but it can be bigger. For example, for files (which is probably what the customer is asking about when they talk about `CreateFile`), the values are defined as follows, in `winnt.h`:

```
#define DELETE                           (0x00010000L)
#define READ_CONTROL                     (0x00020000L)
#define WRITE_DAC                        (0x00040000L)
#define WRITE_OWNER                      (0x00080000L)
#define SYNCHRONIZE                      (0x00100000L)

#define STANDARD_RIGHTS_REQUIRED         (0x000F0000L)

#define STANDARD_RIGHTS_READ             (READ_CONTROL)
#define STANDARD_RIGHTS_WRITE            (READ_CONTROL)
#define STANDARD_RIGHTS_EXECUTE          (READ_CONTROL)

#define FILE_READ_DATA            ( 0x0001 )    // file & pipe
#define FILE_LIST_DIRECTORY       ( 0x0001 )    // directory

#define FILE_WRITE_DATA           ( 0x0002 )    // file & pipe
#define FILE_ADD_FILE             ( 0x0002 )    // directory

#define FILE_APPEND_DATA          ( 0x0004 )    // file
#define FILE_ADD_SUBDIRECTORY     ( 0x0004 )    // directory
#define FILE_CREATE_PIPE_INSTANCE ( 0x0004 )    // named pipe


#define FILE_READ_EA              ( 0x0008 )    // file & directory

#define FILE_WRITE_EA             ( 0x0010 )    // file & directory

#define FILE_EXECUTE              ( 0x0020 )    // file
#define FILE_TRAVERSE             ( 0x0020 )    // directory

#define FILE_DELETE_CHILD         ( 0x0040 )    // directory

#define FILE_READ_ATTRIBUTES      ( 0x0080 )    // all

#define FILE_WRITE_ATTRIBUTES     ( 0x0100 )    // all

#define FILE_ALL_ACCESS (STANDARD_RIGHTS_REQUIRED | SYNCHRONIZE | 0x1FF)

#define FILE_GENERIC_READ         (STANDARD_RIGHTS_READ     |\
                                   FILE_READ_DATA           |\
                                   FILE_READ_ATTRIBUTES     |\
                                   FILE_READ_EA             |\
                                   SYNCHRONIZE)

#define FILE_GENERIC_WRITE        (STANDARD_RIGHTS_WRITE    |\
                                   FILE_WRITE_DATA          |\
                                   FILE_WRITE_ATTRIBUTES    |\
                                   FILE_WRITE_EA            |\
                                   FILE_APPEND_DATA         |\
                                   SYNCHRONIZE)

#define FILE_GENERIC_EXECUTE      (STANDARD_RIGHTS_EXECUTE  |\
                                   FILE_READ_ATTRIBUTES     |\
                                   FILE_EXECUTE             |\
                                   SYNCHRONIZE)
```

Right off the bat, you can see that of the standard rights, `FILE_ALL_ACCESS` includes `STANDARD_RIGHTS_REQUIRED`, whereas the `FILE_GENERIC_*` values include only `STANDARD_RIGHTS_*`, all of which are defined as `READ_CONTROL`. This means that `FILE_ALL_ACCESS` includes `DELETE`, `WRITE_DAC`, and `WRITE_OWNER` which are not included in any of the other generic access masks. (`SYNCHRONIZE` is explicitly added by all of the `FILE_GENERIC_*` access masks.)

If you study it a bit more, you'll see that `FILE_ALL_ACCESS` also includes `FILE_DELETE_CHILD`, which is not present in any of the other generic access masks.

So even in the specific case of file access, we see that `GENERIC_ALL` is not equivalent to `GENERIC_READ | GENERIC_WRITE | GENERIC_EXECUTE`.

Raymond Chen

**Follow**